import streamlit as st

from PIL import Image

import numpy as np

import matplotlib.pyplot as plt

import torch

from torchvision.models.detection import fasterrcnn\_resnet50\_fpn\_v2, FasterRCNN\_ResNet50\_FPN\_V2\_Weights

from torchvision.utils import draw\_bounding\_boxes

weights = FasterRCNN\_ResNet50\_FPN\_V2\_Weights.DEFAULT

categories = weights.meta["categories"] ## ['\_\_background\_\_', 'person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'N/A', 'stopsign',]

img\_preprocess = weights.transforms() ## Scales values from 0-255 range to 0-1 range.

@st.cache\_resource

def load\_model():

    model = fasterrcnn\_resnet50\_fpn\_v2(weights=weights, box\_score\_thresh=0.8)

    model.eval(); ## Setting Model for Evaluation/Prediction

    return model

model = load\_model()

def make\_prediction(img):

    img\_processed = img\_preprocess(img)

    prediction = model(img\_processed.unsqueeze(0))

    prediction = prediction[0]                       ## Dictionary with keys "boxes", "labels", "scores".

    prediction["labels"] = [categories[label] for label in prediction["labels"]]

    return prediction

def create\_image\_with\_bboxes(img, prediction): ## Adds Bounding Boxes around original Image.

    img\_tensor = torch.tensor(img) ## Transpose

    img\_with\_bboxes = draw\_bounding\_boxes(img\_tensor, boxes=prediction["boxes"], labels=prediction["labels"],

                                          colors=["red" if label=="person" else "green" for label in prediction["labels"]] , width=2)

    img\_with\_bboxes\_np = img\_with\_bboxes.detach().numpy().transpose(1,2,0) ### (3,W,H) -> (W,H,3), Channel first to channel last.

    return img\_with\_bboxes\_np

## Dashboard

st.title("Objects Detector :tea: :coffee:")

upload = st.file\_uploader(label="Upload Image :", type=["png", "jpg", "jpeg"]) ## Image as Bytes

if upload:

    img = Image.open(upload)

    prediction = make\_prediction(img)

    img\_with\_bbox = create\_image\_with\_bboxes(np.array(img).transpose(2,0,1), prediction) ## Give image as (3,W,H) Channel first

    fig = plt.figure(figsize=(12,12))

    ax = fig.add\_subplot(111)

    plt.imshow(img\_with\_bbox);

    plt.xticks([],[]);

    plt.yticks([],[]);

    ax.spines[["top", "left", "right", "bottom"]].set\_visible(False)

    st.pyplot(fig, use\_container\_width=True)

    del prediction["boxes"]

    prediction["scores"] = prediction["scores"].detach().numpy().tolist()

    st.header("Prediction Probabilities")

    st.write(prediction)